

---

# Towards Reducing Labeling Cost in Deep Object Detection

---

Ismail Elezi<sup>1\*</sup> Zhiding Yu<sup>2</sup> Anima Anandkumar<sup>2,3</sup> Laura Leal-Taixé<sup>1</sup> Jose M. Alvarez<sup>2</sup>  
<sup>1</sup>TUM <sup>2</sup>NVIDIA <sup>3</sup>CALTECH

## Abstract

Deep neural networks have reached very high accuracy on object detection but their success hinges on large amounts of labeled data. To reduce the dependency on labels, various active-learning strategies have been proposed, typically based on the confidence of the detector. However, these methods are biased towards best-performing classes and can lead to acquired datasets that are not good representatives of the data in the testing set. In this work, we propose a unified framework for active learning, that considers both the uncertainty and the robustness of the detector, ensuring that the network performs accurately in all classes. Furthermore, our method is able to pseudo-label the very confident predictions, suppressing a potential distribution drift while further boosting the performance of the model. Experiments on PASCAL VOC07+12 and MS-COCO show that our method consistently outperforms a wide range of active-learning methods, yielding up to a 7.7% relative improvement in mAP, or up to a 82% reduction in labeling cost.

## 1 Introduction

The performance of deep object detection networks [16, 19] depends heavily on the size of the labeled dataset. Adding more labeled data helps, yet adding more data costs. Motivated by this, researchers have been exploring smart strategies to select the most informative samples in the dataset for labeling, known in the literature as active learning (AL). Sample selection is based on the definition of the acquisition function, which is typically computed by using the network’s uncertainty, thereby selecting to label the samples for which the network is least confident with regard to its predictions.

AL approaches typically come with two problems. The acquisition function is only meaningful if the network is already well-trained for the task, which is not always the case as we might still be in the early cycles of AL. Besides, even if the network performs well in most classes, it might have a low accuracy on a particular class because the class is underrepresented in the dataset or because of intra-class variance. In those cases, using the network predictions to compute the acquisition function for AL often reaches worse performance than using random sampling. Furthermore, by focusing only on the most uncertain samples, the acquired dataset does not contain easy (non-informative) samples, i.e., samples for which the network is certain, thus its distribution might be very different compared to the testing set. In order to create a balanced dataset, we need to include easy samples. Nonetheless, naively including them in the standard AL cycle would be a waste of resources as we would be labeling samples for which the network is certain.

In this work, we propose an active learning framework to address the aforementioned problems. There are two key components in our framework: a robustness-based acquisition function for hard samples and a pseudo-labeling scheme for easy samples. The acquisition function is based on the *robustness* of the network, using a *consistency* loss and therefore is class-agnostic, consequently is reliable also for objects of classes where the network is not accurate. Concretely, we feed to the network images and their augmented versions, e.g., by doing horizontal flipping, and train the

---

\*Work performed while interning at NVIDIA.

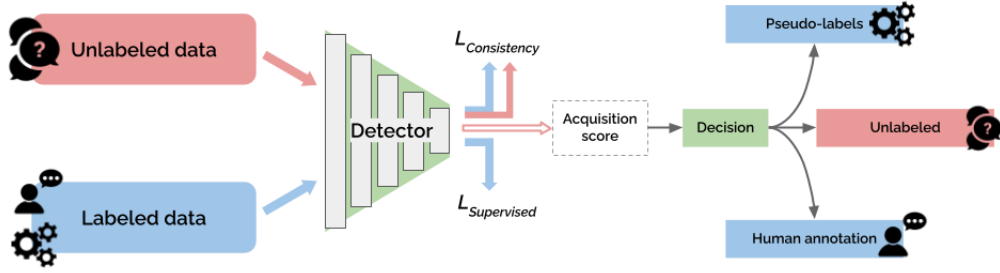


Figure 1: Overview of our method. We first train the network in a semi-supervised manner. During active learning, for each image, we use the network to compute the acquisition function and based on it decide if we actively label the image, if we pseudo-label it, or if we only use it as part of the unlabeled data for the next training cycle.

network to output *consistent* predictions [9]. We then compute the difference between the predictions of the original images and their augmented version, ignoring their correctness. We use the same consistency loss as the acquisition function during AL. Empirically, we show that this method does not suffer from the drawbacks of traditional AL approaches.

This acquisition function is biased towards the most informative (hard) samples, consequently, the resulting dataset will no longer represent the real distribution at test time. In order to incorporate easy samples to compensate for potential dataset distribution drifts without additional labeling costs, we propose a pseudo-labeling scheme. Hence, for every active learning cycle, we use the previously trained network to mine easy samples, i.e., samples where the network is confident about its prediction, and use the network’s own prediction as pseudo-labels. These labels will be used to train the network in the next AL cycle. As this is an auto-labeling process, it incurs no extra labeling cost, allowing us to spend all the labeling budget on the hard samples.

In summary, our **contributions** are the following:

- We propose a novel class-agnostic active learning score based on the *robustness* of the network, using a novel *consistency* score.
- We add a pseudo-labeling module in order to leverage the less informative samples, expanding the labeled dataset, without incurring in extra human annotation costs.
- We demonstrate the benefits of our method in two publicly available datasets: PASCAL VOC07+12 and MS-COCO. Compared to state-of-the-art active learning methods [21, 26], our approach yields up to a 7.7% and 7% relative mAP improvement for PASCAL-VOC and MS-COCO, respectively. Importantly, we can achieve the same performance as the baseline but reduce up to 82% of the labeling costs.

## 2 Related Work

**Deep Active Learning for Object Detection.** *Deep Active Learning* (AL) has sparked the interest of researchers in the last few years. The work of [2] trains an ensemble of neural networks and then selects the samples with the highest score defined by some acquisition function, i.e., entropy [22], or BALD [8]. Concurrent works [6, 12] explore a similar direction by approximating the uncertainty via Monte-Carlo dropout [5]. The work of [2] compares the approaches, decisively concluding that the ensemble approach reaches higher results at the cost of more computational power. Another Bayesian approach [25] trains a variational autoencoder (VAE) [11] on both real and augmented samples, and then chooses to label the samples with the highest reconstruction error. A different approach is the core-set [21] that chooses to label a set of points such that a model trained over the selected subset is competitive for the remaining data points. In general, all these methods were initially designed for classification-based active learning and then applied to the object detection task.

More recently, several methods have been tailored specifically for the task of object detection. The work of [1] proposes a solution by training a network that computes dense object prediction probabilities for each unlabeled image, followed by computing pixel-scores and aggregating them into a frame-level score. A different solution was given by [10], where the authors define two different

scores: localization tightness, which is the overlapping ratio between the region proposal and the final prediction, and localization stability, based on the variation of predicted object locations when input images are corrupted by noise. In all cases, the images with the highest scores are chosen to be labeled. The work of [20] proposes a "query by committee" paradigm to choose the set of images to be queried. Another approach is that of [3] where instead of directly querying bounding box annotations (strong labels) for the most informative samples, they first query weak labels and optimize the model. Then, using a switching condition, the required supervision level is increased. Yet another approach has been proposed in [7], where an ensemble of object detectors provides potential bounding boxes and probabilities for each class of interest. Then, a scoring function is used to obtain a single value representing the informativeness of each unlabeled image. The work of [26] gives a heuristic but elegant solution while reaching state-of-the-art results compared with other single-model methods. The authors train a network in the task of detection while learning to predict the final loss. In the sample acquisition stage, samples with the highest prediction loss are considered the most interesting ones and are chosen to be labeled.

**Semi-Supervised Learning for Object Detection.** *Deep Semi-Supervised Learning (SSL)* is a deep learning approach that combines a small amount of labeled data with a large amount of unlabeled data during neural network training. Unlike in AL, where the unlabeled data is used only during the acquiring stage, in SSL, the unlabeled data is an important part of the training stage. Several methods have shown excellent results [13, 14, 17, 24] by casting the problem of semi-supervised learning as a regularization problem, in effect adding a new loss for the unlabeled samples. With the methods using different datasets, dataset splits, network backbones, and hyperparameters, it was not clear which strategies really performed best. The comprehensive survey of [18] addresses this issue with an intensive evaluation of several methods under the same settings.

For object detection, a simple but efficient semi-supervised method was introduced in [9]. The authors developed a consistency-based model, where each image is horizontally flipped, with the loss function being defined as the prediction inconsistency between the original image and its flipped version. The method significantly improved the mAP score in Pascal VOC dataset [4].

In this work, instead of developing an acquisition function that is only based on the uncertainty of the detector, we devise an acquisition function that combines the uncertainty of the detector with its robustness, considering all the classes in the dataset. We then add a pseudo-labeling module that labels the easy images for free. Together with our AL method, this ensures that our acquired dataset is a good representative of the original dataset.

### 3 Method

In this section, we present our active learning framework for object detection. Let  $D$  be a dataset divided into a labeled set  $L$  and a pool of unlabeled data  $U$ . We start with a deep object detection network trained on the losses specified in Sec. 3.1, and then proceed with the active learning cycles. These include mining a subset of samples from the pool of unlabeled data  $U$  and transferring them to the labeled set  $L$ , incurring a labeling cost. The proposed acquisition function used to select the samples to label is defined in Sec. 3.2. Intuitively, we are interested in mining hard samples for which we can rely on supervised learning during training. Nonetheless, arbitrarily augmenting the set  $L$  with only hard samples creates a distribution drift in our training data. Hence, we propose to include in training the easy samples, i.e., objects for which the network's confidence is high, by using pseudo-labeling. We train our network with our new set of labeled images, and repeat the whole procedure for  $T$  active learning cycles. To train the object detection network we combine the standard supervised Multibox loss [16] for labeled samples with a semi-supervised consistency loss [9] for unlabeled samples. Fig. 1 and Algorithm 1 show a high-level description of the pipeline and the algorithmic steps, respectively. Below, we give the details of every step.

#### 3.1 Deep Object Detection Training

**Notation.** Let  $\Delta$  be the object predictions for an image, and let  $\Delta_i$  be its  $i$ -th object prediction.  $\Delta_i$  consists of the bounding box  $b_i$  and the class prediction  $c_i$  that represents the probability distribution after the softmax layer of the neural network. The bounding box  $b_i$  consists of the displacement of the center and scale coefficients, represented by the tuple  $[x0, y0, w, h]$ . Given an

augmented version of the original image, e.g., by doing a horizontal flip, we define  $\hat{\Delta}$  to be the set of its object predictions, and  $\hat{\Delta}_i$  consisting of the bounding box  $\hat{\mathbf{b}}_i$  and class  $\hat{\mathbf{c}}_i$ , its  $i$ -th prediction.

**Multibox loss for labeled samples.** For the labeled images, the network is trained with the standard MultiBox loss for class predictions, and a smooth  $L1$  loss for bounding box predictions. Given the network’s class predictions  $\mathbf{c}$  and an indicator  $\mathbf{y}_{ij}^p = \{0, 1\}$  for matching the  $i$ -th box to its corresponding  $j$ -th ground truth box of category  $p$ , the MultiBox loss is defined as:

$$\mathcal{L}_{conf}(\mathbf{c}, \mathbf{y}) = - \sum_{i \in Pos} \sum_{p=1}^{|\text{classes}|} \mathbf{y}_{ij}^p \log(\mathbf{c}_i^p) - \sum_{i \in Neg} \log(\mathbf{c}_i^0), \quad (1)$$

where  $Pos$  define positive bounding boxes (containing objects),  $Neg$  defines bounding boxes of class *background* and  $p$  defines the  $p$ -th category.

**Consistency loss for unlabeled samples.** This loss aims at leveraging the entire dataset including the unlabeled images. During training, we feed an image and its augmented version to an object detector. In our case, we use horizontal flip as augmentation. Given the sets of predictions for the original and augmented image, we first need to match the predictions  $\Delta$  with  $\hat{\Delta}$ . Considering that augmented images are flipped horizontally, we apply a negation to the  $\hat{\mathbf{x}}\mathbf{0}_i$  coordinate of these predictions. We then compute intersection over union (IoU) to match the predictions coming from the two images:

$$\Delta'_i = \operatorname{argmax}_{\mathbf{b}_i \in \{\mathbf{b}\}} \operatorname{IoU}(\mathbf{b}_i, \hat{\mathbf{b}}_i). \quad (2)$$

For each matched pair,  $\Delta'_i$  and  $\hat{\Delta}_i$ , we define their class consistency loss as:

$$\mathcal{L}_{con_C}(\mathbf{c}'_i, \hat{\mathbf{c}}_i) = \frac{1}{2} [\mathcal{KL}(\mathbf{c}'_i, \hat{\mathbf{c}}_i) + \mathcal{KL}(\hat{\mathbf{c}}_i, \mathbf{c}'_i)], \quad (3)$$

where  $\mathcal{KL}$  represents the Kullback-Leibler divergence.

We define the consistency localization loss by computing the euclidean distance between predictions:

$$\mathcal{L}_{con_L}(\mathbf{b}'_i, \hat{\mathbf{b}}_i) = \frac{1}{4} (\|\mathbf{x}\mathbf{0}'_i - (-\hat{\mathbf{x}}\mathbf{0}_i)\|^2 + \|\mathbf{y}\mathbf{0}'_i - \hat{\mathbf{y}}\mathbf{0}_i\|^2 + \|\mathbf{w}'_i - \hat{\mathbf{w}}_i\|^2 + \|\mathbf{h}'_i - \hat{\mathbf{h}}_i\|^2). \quad (4)$$

We compute the total consistency loss by averaging the losses from all matched pairs of predictions:

$$\mathcal{L}_{con} = \mathbb{E}[\mathcal{L}_{con_C}(\mathbf{c}', \hat{\mathbf{c}})] + \mathbb{E}[\mathcal{L}_{con_L}(\mathbf{b}', \hat{\mathbf{b}})]. \quad (5)$$

**Overall Training loss.** Finally, to train the deep detection network, we aggregate the multibox,  $L1$  and consistency losses as:

$$\mathcal{L}_{total} = \mathcal{L}_{con} + \mathcal{L}I + \mathcal{L}_{conf}, \quad (6)$$

where  $\mathcal{L}_{conf}$  is used in all samples, while  $\mathcal{L}_{con}$  and  $L1$  are used in the labeled samples.

### 3.2 Mining data

**Acquisition function for active learning.** Most AL methods use some measure of uncertainty, e.g., the entropy, to compute the acquisition function. A prediction that has a high entropy suggests that the object is highly dissimilar to the already labeled images, thus, if labeled, will provide different information to the ones we have. However, we empirically find that using only an *uncertainty*-based acquisition function is not an ideal solution, especially for images coming from classes where the network has poor performance. If the network’s predictions for a class are incorrect, as we show in the experiments, they are also unreliable to compute the acquisition function.

In this paper, we re-purpose the classification *consistency* loss  $\mathcal{L}_{con_C}$  as an acquisition function. Intuitively, if an object which was part of the consistency loss-based training still yields a low consistency value, continuing training the network with this object in the same loss will likely not benefit the network. In contrast, labeling that object will enable it to be used in the supervised part. Importantly, this consistency-based score is a general and class-agnostic value that represents the *robustness* of the network. Therefore, it is complementary to the uncertainty-based scores.

We now describe our process to select data for labeling. First, we feed to the network every image in the acquisition pool  $U$  and apply non-maximum suppression. An image has typically several

---

**Algorithm 1:** Towards reducing labeling costs in deep object detection

---

**Input:** Set of labeled images  $L$ , acquisition pool of unlabeled images  $U$ , neural network  $\phi_0$ , active learning budget  $N$ , number of active learning cycles  $T$ , pseudo-label threshold  $\tau_2$ .

- 1 Use the initial labeled data to train network  $\phi_0$  using MultiBox loss and consistency loss as given on Equation 6.
  - 2 **for**  $it = 1, \dots, T$  **do**
  - 3     1) Sort all the images based on their acquisition function that considers the robustness, uncertainty and the confidence, as described on Equation 7.
  - 4     2) Select the  $N/T$  images with the highest score and transfer them from  $U$  to  $L$ .
  - 5     3) Use  $\phi_{it-1}$  to pseudo-label the images coming from  $U$  that contain predictions with higher confidence than  $\tau_2$  and use those labels in the next cycles.
  - 6     4) Train the network  $\phi_{it}$  using the modified MultiBox loss, as given on Equation 8, as well as the consistency loss.
- 

predicted bounding boxes representing the objects contained in the image. Intuitively, labeling an image that has at least one *difficult* object, independently of the number of *easy* objects is beneficial because of the difficult object. However, choosing to label only images that contain difficult objects could lead to a distribution drift as those objects might not be good representatives of the dataset. For the object  $\Delta_i$ , we formulate our acquisition score as:

$$A(\Delta_i) = H(\Delta_i) \times I(\Delta_i) \text{ conditioned on } C(\Delta_i) \geq \tau_1, \quad (7)$$

where  $C(\Delta_i)$ ,  $H(\Delta_i)$ , and  $I(\Delta_i)$  represent the confidence, entropy and inconsistency of  $\Delta_i$ , and  $\tau_1$  is a fixed threshold to condition the scoring on predictions whose confidence is equal or larger than the threshold. That is, we are scoring only moderately-difficult objects. We then define the acquisition function for  $\Delta$  as the maximum score of all its objects  $\Delta_i$ . We then sort all the images based on their acquisition score and select to label the  $N/T$  images with the highest score, where  $N$  corresponds to the acquisition budget, and  $T$  corresponds to the number of active learning steps. We repeat this procedure for  $T$  steps. Note that we annotate every bounding box that belongs to a *selected image* regardless if the box has a high score or not.

**Pseudo-labeling to prevent distribution drift.** The active learning pipeline described above targets the hard samples, ignoring the easier samples. While not targeting the very hard samples helps, we argue that the network should see some representative data in order to ensure that no distribution drift happens. At the same time, we want to avoid labeling easy samples to not spend labeling resources. Hence, we propose to add a pseudo-labeling module where the network trained in the previous active learning cycle provides pseudo-labels for the network that is being trained on this cycle. We pseudo-label only the bounding boxes where the network is confident, i.e., the confidence is above some threshold  $\tau_2$ . Finally, we discretize these predictions, setting them to hard labels, to use them as ground truth during the training of the current network.

We now add the loss for the pseudo-labeling part. We note that in an image, the network might be confident for some predicted bounding box, and not confident for the others. In this case, we pseudo-label only the confident box. Considering that the network might still make predictions for the parts of the image where there is no pseudo-label, we must change the loss function to not penalize these predictions. Thus, we can rewrite the MultiBox Loss function as:

$$\mathcal{L}_{conf}(c, y, \hat{y}) = - \sum_{i \in Pos} \sum_{p=1}^{|classes|} y_{ij}^p \log(c_i^p) - \sum_{i \in Neg} \log(c_i^0) - \sum_{i \in Pos} \sum_{p=1}^{|classes|} \hat{y}_{ij}^p \log(c_i^p), \quad (8)$$

where  $\hat{y}$  and  $\hat{Pos}$  represent the indicator and the positive bounding boxes for the pseudo-labels.

## 4 Experiments

In this section, we demonstrate the effectiveness of our approach to improve the performance of object detection. For all experiments, we report mean average precision (mAP) as main metric, and use two public datasets: PASCAL VOC07+12 (VOC07+12) [4] and MS-COCO train2014 [15].

VOC07+12 consists of 16,551 images for training, and 4,952 testing images taken from VOC07 testset. MS-COCO consists of 83K images for training, and MS-COCO valset2017 contains 5,000 images for testing.

Following [26], on VOC07+12 we start by randomly sampling 2,000 images. On the larger MS-COCO, we start by randomly sampling 5,000 images. In each case, we perform 5 active learning cycles, and in each cycle, we choose 1,000 images for labeling. To ensure that the network does not diverge, we define each mini-batch to have half the images labeled. We set the confidence threshold for AL to  $\tau_1 = 0.5$ , and the pseudo-label threshold to  $\tau_2 = 0.99$  based on the results of the zeroth active learning step in VOC07+12.

For a fair comparison with [26], we use the Single-Shot Detector 300 (SSD300) [16] based on a VGG [23] backbone for all our experiments. We train the model for 120,000 iterations using SGD with momentum. We set the initial learning rate to 0.001 and divide it by 10 after 80,000 and 100,000 iterations, respectively. We use batches of size 32 and a constant L2 regularization parameter set to 0.0005. For fair comparison with [26], we use the same model, hyperparameters and the same public implementation<sup>2</sup>. We train all networks using four NVIDIA V100 GPUs. In all experiments, we train three independent networks using the same initial split of randomly sampled images and report the mean. We give the exact numbers of the mean and standard deviation in the supplementary material.

#### 4.1 Comparison with other methods

We first compare our results with other active learning methods. Specifically, we use two baselines, random and entropy sampling, and two state-of-the-art methods using a single model, namely, Coreset [21] and Learning Loss [26]. We also include results of two-multimodel approaches, namely, MC-dropout [6] and ensemble-based [2] active learning (consisting of three neural networks). Finally, we compare to the semi-supervised learning method proposed in [9] and a pseudo-labeling method [14]. We use the publicly available code for all existing methods except for Learning Loss [26]. In this case, as there is no public implementation, we only provide numbers for VOC07+12 as reported in the original publication.

In Fig. 2, we show the results of this comparison for both datasets. For VOC07+12, in Fig. 2a, we observe that starting from the first two active learning cycle, our method has a relative improvement over the random baseline by 10.5%, over the best overall active learning method [21] by 7.7%, and it outperforms the semi-supervised method of [9] by 5.6%. We see that the performance improvement of our method is maintained in the other active learning cycles. In the last one, where we use 7,000 samples, 5,000 of which are actively sampled, our method outperforms the random baseline by 9.1%, best existing active learning methods by more than 5.7% [26], and the semi-supervised learning approach by 3.4%. Multi-model active learning networks, namely, ensemble [2] or MC-dropout [6] typically outperform single models at the cost of longer training and active learning time, and in the case of the ensemble has 3 times more training parameters. Nonetheless, our proposed single model still reaches better results than multi-model methods, outperforming the ensembles by 8% in the first AL cycle, and 1.8% in the last cycle.

For MS-COCO, in Fig. 2b, we observe that for the first active learning cycle, our method outperforms the random baseline by 5.8%, the best-performing AL method by 5.3% [22], the semi-supervised method by 5.4%, and the ensembles by 5%. In the second cycle, our approach outperforms all the other methods, including ensembles, by 6.3% or more. We observe that this difference is maintained in the other cycles, including the last active learning cycle where our method outperforms the semi-supervised method [9], multi-model methods [2, 6] and the best AL method [21] by 3%.

#### 4.2 Ablation studies

**The effect of each module.** We first focus on validating the effect of each component of our framework on the final performance. In Fig. 4, we present the performance comparison of the semi-supervised model on VOC07+12 under different acquisition functions (random, entropy, inconsistency) and two instances of our method: with and without using pseudo-labels. We see that on the first active learning cycle, neither entropy nor inconsistency significantly outperforms the results of random sampling. However, we immediately see a significant effect, i.e., a relative improvement over 0.9% using entropy and 1.4% using inconsistency, in the second active learning cycle. We see that

<sup>2</sup><https://github.com/amdegroot/ssd.pytorch>

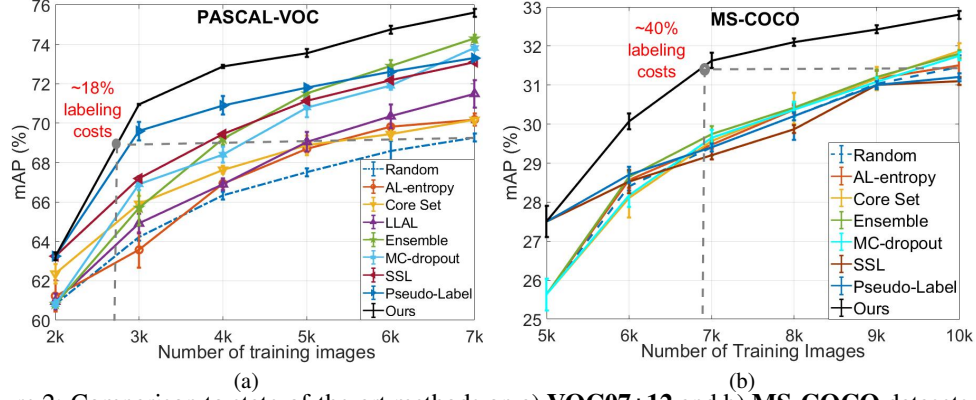


Figure 2: Comparison to state-of-the-art methods on a) **VOC07+12** and b) **MS-COCO** datasets. Our method outperforms all the other methods, including ensembles, by a large margin in both datasets.

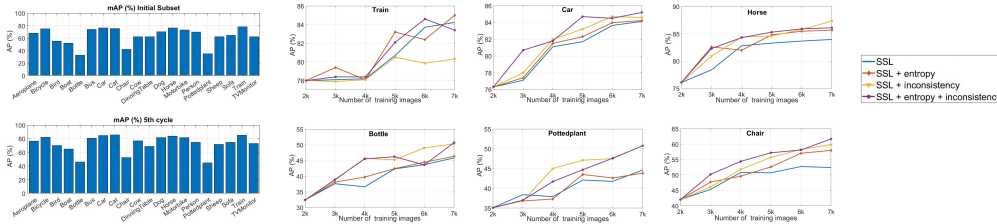


Figure 3: **VOC07+12**: In the bar plots we show the accuracy per class using random sampling in the zeroth and last cycle. We present the results of each AL method for the three best-performing ("Train", "Car", and "Horse") and worst-performing ("Bottle", "Pottedplant", and "Chair") classes.

the increase in performance gets bigger in the next active learning cycle and, in fact, in the fifth active learning cycle, the performance gain from the entropy is 2.3% and from inconsistency is 2.4%.

We then show the results of our acquisition function. In the first active learning cycle, we immediately see a significant improvement in performance. While entropy (67.24mAP) and inconsistency (67.39mAP) reach an insignificant improvement over random sampling (67.19mAP), our acquisition function reaches 68.40mAP, which is 1.5% better than random sampling. The performance improvement gets larger in the next cycles: 2% in the second cycle, 2.5% in the third cycle, and a peak improvement of 2.6% in the fifth active learning cycle. In all cases, our proposed score outperforms both active learning methods that are based on a single acquisition function.

We further study the effect of pseudo-labeling in our framework. In Fig. 4, we observe that on the first active learning cycle, adding pseudo-labels comes with an immediate boost, improving the results by 2% compared to the already well-performing acquisition function for semi-supervised learning (3.9% better than the semi-supervised method that uses random sampling). We further observe that on the second cycle, it gives an improvement of 1.5% compared to using only our acquisition function (3.5% better than the semi-supervised method that uses random sampling). The Pseudo-labeling module continues to give a boost in performance in all the following AL steps, although we see saturation in the performance boost.

We then evaluate our method when we exclude the outliers, the predictions for which the network's confidence is lower than  $\tau_1 = 0.5$ . We see that this improves our results by a further 1.5% in the first AL cycle and by 1.4% in the second AL cycle. The performance gain in the later iterations gets lower, nevertheless, we still improve the results by 0.3% in the fifth AL cycle.

In Fig. 4b, we provide a similar ablation study for MS-COCO. We again observe that the combined score outperforms both the entropy and inconsistency scores in isolation. However, unlike in VOC07+12, we observe that using only the entropy, the improvement is marginal over random sampling. On the other hand, we observe that inconsistency works significantly better than random sampling and entropy (we provide an intuitive explanation in the next section). We further observe the effect of pseudo-labels. We see that in the first AL cycle, adding pseudo-labels boosts the performance by 3.1% and the performance boost is maintained up to the last cycle.

**Acquisition functions.** We now focus on analyzing the effects of aggregating the two acquisition functions. To do so, in Fig. 3, we check the performance of every individual class in the zeroth and the last AL cycle. We then focus on the three best-performing classes ("*Train*", "*Car*", and "*Horse*") and the three worst-performing classes ("*Bottle*", "*Pottedplant*", and "*Chair*"). A first observation is that for the best-performing classes, entropy-based AL, on average, tends to outperform inconsistency-based AL. We also see that while the inconsistency score does a good job in classes "*Car*" and "*Horse*", it entirely fails on the best performing class "*Train*", performing worse than the random sampling.

On the other hand, we see that inconsistency-based AL outperforms the entropy-based AL by a significant margin in all three worst-performing classes. While the entropy-based AL on average seems to only slightly outperform random sampling, the inconsistency-based AL gives a relative performance gain of up to 24%, 14% and 18% in classes "*Bottle*", "*Pottedplant*", and "*Chair*". Intuitively, one can argue that this phenomenon is actually to be expected. The fact that the network does a poor job on its predictions leads to its class predictions being unreliable for any uncertainty-based AL method. At the same time, a more general acquisition function that is dependent only on the robustness of the network is better suited for low-performing classes. This explains why in MS-COCO, which contains many more challenging classes, the inconsistency significantly outperforms the entropy. Finally, we show that our acquisition function reaches the best results.

**Ratio of pseudo-labels.** We now study the effect of increasing the number of pseudo-labels by allowing more noisy pseudo-labels. To do so, we lower the pseudo-labeling threshold  $\tau_2$  from 0.99 to 0.9 and 0.5. We present the results in Fig. 5a. We observe that we reach the best overall results by using an extremely high threshold  $\tau_2 = 0.99$ . Decreasing  $\tau_2$  to 0.9 and thus allowing more pseudo-labels harms the performance. Further decreasing it to 0.5, hence allowing many more pseudo-labels, actually harms the entire training. We thus conclude that we need to be selective in the choice of pseudo-labels.

To understand why the performance improvement of the network trained with the pseudo-labels module diminishes in the later active learning cycles, we study the pseudo-labels gain as a function of the pseudo-labels ratio to the entire labels. As we show in Fig. 5b, in the first active learning cycle where the pseudo-labels bring a maximum gain (3.7%), roughly half of the labels are pseudo-labels. With the decrease of the number of pseudo-labels, we see a tendency for the gain to lower. Our intuition is that when the number of pseudo-labels is high, despite them being noisy, they still help the training process. However, when the number of pseudo-labels gets lower, their effect gets smaller. Note that for MS-COCO, where the number of images that can be potentially pseudo-labeled is much higher (78K compared to 14, 651 in PASCAL VOC), the performance gain from the pseudo-labels module does not diminish. Intuitively, this is explained by the fact that the ratio of pseudo-labels in all cycles remains high.

**Performance boost per class.** We now study if the pseudo-labels help only some particular classes, or if they help in all classes. In Fig. 6, we plot the performance gain coming from the module for each class and compare it with the performance gain from AL alone, and random sampling. In the first AL cycle, we see that pseudo-labels improve over random sampling in all 20 classes, with AL alone gives a negative boost in two classes: "*Pottedplant*" and "*DiningTable*". Furthermore, they outperform AL alone in 17 out of 20 classes. We also find out that pseudo-labels give a boost over AL alone in all three worst-performing classes ("*Bottle*", "*Pottedplant*", and "*Chair*"). We see a

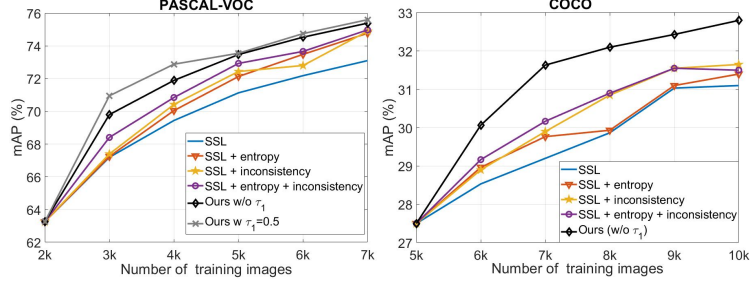


Figure 4: Accuracy as a function of the acquisition function. Ablation study on the effect of entropy, consistency, combined combined score with pseudo-labeling, combined score with pseudo-labeling conditioned in  $\tau_1$  as active learning methods for our semi-supervised network. In the x-axis we have the number of actively-labeled samples, in the y-axis we show the mAP score of detection.



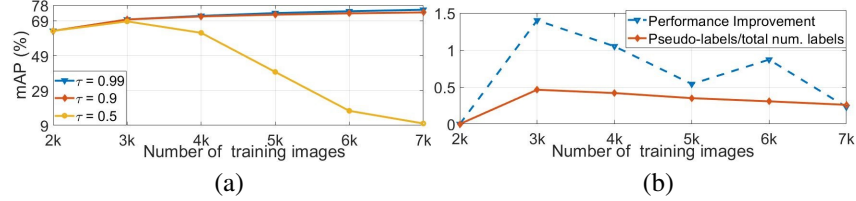


Figure 5: **VOC07+12**: a) Accuracy as a function of  $\tau_2$  for selecting pseudo-labels. b) Accuracy improvement with respect to the pseudo-labels ratio to the entire labels.

similar pattern in the other cycles. In the second cycle, the pseudo-labels module improves over AL alone in 14 classes and gives a negative boost in only one class ("Dog") compare to AL alone that gives a negative boost in four classes. In the third cycle, the pseudo-labels module improves over AL alone in 11 classes and gives a negative boost in two classes (compared to AL in five classes). In the fourth cycle, the pseudo-labels module improves over AL alone in 15 classes and gives a negative boost in one class (compared to AL in three classes). Finally, in the fifth AL cycle, the pseudo-labels module improves over AL alone in 10 classes with class "Car" being a tie and gives a negative boost in only one class, compared to AL in three classes. We thus conclude that by focusing only on the *hard* samples, AL alone harms the performance on several classes. However, adding pseudo-labels diminishes this effect, making the network much more robust and thus preventing a dataset drift.

## 5 Conclusions

In this work, we developed a framework that reduces the labeling costs for object detection. Our framework consists of a novel acquisition function that is based on the *robustness* of the neural network with respect to its predictions. We show that our acquisition function works particularly well on low-performing classes. We then show that our score can be naturally combined with uncertainty-based scores such as entropy to boost the performance of AL in all classes. Furthermore, we show that adding a pseudo-label module for the easy samples nicely complements our acquisition function and prevents a potential distribution drift. In this way, our unified framework chooses to actively label the most informative samples in the dataset, while it pseudo-labels the easiest samples. This allows us to use the majority of the dataset in a supervised manner, while reducing the labeling costs. As we showed in the experimental setting, we reach the same results as a fully-supervised baseline, but by reducing the labeling costs by up to 82%.

A limitation of our method is that it cannot work with unknown classes in an open-world setting. Furthermore, our method is task-specific and thus less suitable for acquiring datasets for multi-task networks. Finally, we only experimented with a constant pseudo-labeling threshold, which might not be optimal. Future work will consist of addressing the above-mentioned issues, with a focus on making our framework suitable for the open-world setting.

## Broader Impact

Our work introduces a unified framework for reducing the labeling costs needed to train object detection networks. It provides a way of using all the samples in the dataset, be they labeled or not, in an optimized way to reach high accuracy. Manually selecting and labeling frames takes a tremendous amount of time and labor, so by selecting the right data for annotation and training, our approach can positively impact by reducing storage and labeling costs on industries such as autonomous driving that require large amounts of labeled data. Our approach uses a single model with the minimum amount of training data to maximize performance from an ecological standpoint. Our results suggest that this is a practical approach to address inefficiencies in training data selection for real-world applications such as autonomous driving. As our approach requires fewer training resources, thus we also reduce the carbon footprint.

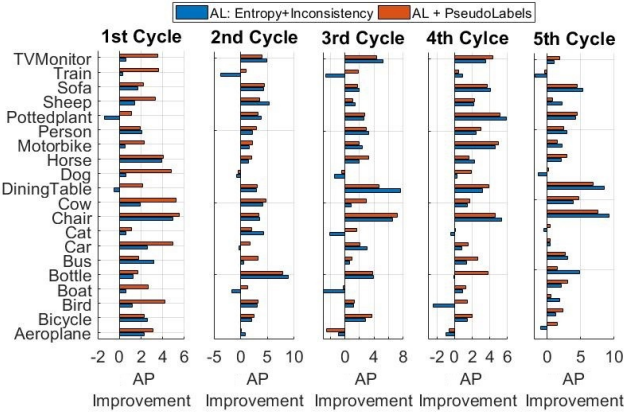


Figure 6: **VOC07+12**: Effect of pseudo-labels compared to AL alone for every class.

# Supplementary Material

## Exact numbers for the experiments given in the main paper

In the paper, we provide plots for the main experiments due to the limited space. Here, in Tables 1, 2, 3, 4 we summarize the exact numbers corresponding to Figures 2a, 2b, 4a, 4b of the main paper, respectively. We provide the mean and the standard deviation for each method and AL cycle. Every experiment has been run three times.

Cycle	Random	Entropy	Core-Set	LLAL	Ensemble	MC-dropout	SSL	PL	Ours
0	60.82±0.2	61.23±0.8	62.36±0.5	60.95±0.4	60.82±0.2	60.82±0.2	<b>63.25±0.2</b>	<b>63.25±0.2</b>	<b>63.25±0.2</b>
1	64.23±0.2	63.57±0.9	65.90±0.4	64.91±0.5	65.70±0.9	66.90±0.3	67.19±0.1	69.6±0.5	<b>70.95±0.1</b>
2	66.33±0.2	66.94±0.2	67.63±0.2	66.90±0.3	69.20±0.3	68.40±0.2	69.44±0.1	70.9±0.5	<b>72.88±0.1</b>
3	67.51±0.2	68.70±0.2	68.88±0.5	69.05±0.5	71.50±0.2	70.80±0.4	71.13±0.1	71.8±0.1	<b>73.55±0.2</b>
4	68.60±0.5	69.82±0.1	69.44±0.3	70.35±0.6	72.90±0.3	71.90±0.5	72.18±0.1	72.6±0.2	<b>74.75±0.2</b>
5	69.27±0.2	70.18±0.3	70.16±0.1	71.49±0.7	74.29±0.2	73.81±0.0	73.1±0.1	73.3±0.1	<b>75.60±0.2</b>

Table 1: **VOC07+12**. Comparison to state-of-the-art methods. We initially use 2,000 randomly sampled images and, in every other cycle, we label 1,000 extra images. Our method outperforms all the other methods, including ensembles, by a large margin.

Cycle	Random	Entropy	Core-Set	Ensemble	MC-dropout	SSL	PL	Ours
0	25.63±0.4	25.63±0.4	<b>25.63±0.4</b>	25.63±0.4	25.63±0.4	<b>27.50±0.3</b>	<b>27.50±0.3</b>	<b>27.50±0.3</b>
1	28.40±0.1	28.57±0.2	28.10±0.5	28.65±0.1	28.17±0.3	28.53±0.1	28.70±0.2	<b>30.07±0.4</b>
2	29.40±0.2	29.47±0.1	29.57±0.1	29.75±0.2	29.65±0.2	29.20±0.1	29.42±0.2	<b>31.63±0.1</b>
3	30.20±0.6	30.37±0.1	30.40±0.4	30.43±0.1	30.37±0.2	29.87±0.1	30.24±0.2	<b>32.10±0.1</b>
4	31.03±0.1	31.17±0.2	31.17±0.3	31.20±0.2	31.09±0.1	31.03±0.1	31.03±0.1	<b>32.43±0.1</b>
5	31.47±0.3	31.50±0.2	31.87±0.2	31.75±0.1	31.80±0.1	31.10±0.1	31.22±0.1	<b>32.80±0.0</b>

Table 2: **MS-COCO**. Comparison to state-of-the-art methods. In this case, we initially use 5,000 randomly sampled images, and, in every active learning cycle, we label 1,000 extra images. Our method outperforms all the other methods, including ensembles, by a large margin.

Cycle	Random	Entropy	Inconsistency	Combined	Ours w/o $\tau_1$	Ours
0	<b>63.25±0.2</b>	<b>63.25±0.2</b>	<b>63.25±0.2</b>	<b>63.25±0.2</b>	<b>63.25±0.2</b>	<b>63.25±0.2</b>
1	67.19±0.1	67.24±0.1	67.39±0.9	68.40±0.3	69.80±0.1	<b>70.95±0.1</b>
2	69.44±0.1	70.05±0.1	70.42±0.6	70.84±0.8	71.89±0.1	<b>72.88±0.1</b>
3	71.13±0.1	72.13±0.2	72.43±0.4	72.93±0.3	73.47±0.1	<b>73.55±0.2</b>
4	72.18±0.1	73.48±0.6	72.80±1.0	73.66±0.2	74.53±0.2	<b>74.75±0.2</b>
5	73.1±0.1	74.77±0.2	74.90±0.3	74.98±0.5	75.39±0.1	<b>75.60±0.2</b>

Table 3: Accuracy as a function of the acquisition function. Ablation study on the effect of entropy, consistency, combined score, combined score with pseudo-labeling but without conditioning in  $\tau_1$ , and the overall framework in VOC07+12. We observe that doing active learning with either entropy or consistency outperforms the semi-supervised model, that the combined score performs better than either of the individual scores and that adding pseudo-labels in combination with the combined score reaches the best overall results. **Ours** refers to our approach combining both acquisition functions, pseudo-labeling and conditioning in  $\tau_1$ .

## Pseudo-labels for class

In this final experiment we analyze different methods for obtaining pseudo-labels. Precisely, instead of obtaining pseudo-labels using the confidence score and a threshold  $\tau_2$  independently of the class, we consider the  $k\%$  most confident objects for each class and add them as pseudo-labels. In Table 5 we present the results where we pseudo-label the top 20%, top 30%, top 40% most confident

Cycle	Random	Entropy	Inconsistency	Combined	Ours w/o $\tau_1$
0	<b>27.50</b> $\pm$ 0.3	<b>27.50</b> $\pm$ 0.3	<b>27.50</b> $\pm$ 0.3	<b>27.50</b> $\pm$ 0.3	<b>27.50</b> $\pm$ 0.3
1	28.53 $\pm$ 0.1	28.96 $\pm$ 0.3	28.90 $\pm$ 0.3	29.17 $\pm$ 0.2	<b>30.07</b> $\pm$ 0.4
2	29.20 $\pm$ 0.1	29.77 $\pm$ 0.1	29.90 $\pm$ 0.7	30.17 $\pm$ 0.2	<b>31.63</b> $\pm$ 0.1
3	29.87 $\pm$ 0.1	29.93 $\pm$ 0.4	30.85 $\pm$ 0.4	30.90 $\pm$ 0.2	<b>32.10</b> $\pm$ 0.1
4	31.03 $\pm$ 0.1	31.10 $\pm$ 0.1	31.55 $\pm$ 0.1	31.55 $\pm$ 0.1	<b>32.43</b> $\pm$ 0.1
5	31.10 $\pm$ 0.1	31.40 $\pm$ 0.1	31.60 $\pm$ 0.1	31.65 $\pm$ 0.1	<b>32.80</b> $\pm$ 0.0

Table 4: Accuracy as a function of the acquisition function. Ablation study on the effect of entropy, consistency, combined and combined score with pseudo-labeling as active learning methods for our self-supervised network. We observe that doing active learning with either entropy or consistency outperforms the semi-supervised model, that the combined score performs better than either of the individual scores and that adding pseudo-labels in combination with the combined score reaches the best overall results. **Ours** refers to our approach combining both acquisition functions and pseudo-labeling.

predictions for class and compare them with the results of our method described in the main paper. We see that in general, the methods where we pseudo-label per class work well, in part with our method. However, for simplicity, we choose to use our method that is class-agnostic when it comes to pseudo-labels.

Cycle	Top 20%	Top 30%	Top 40%	Ours w/o $\tau_1$
0	63.25 $\pm$ 0.3	63.25 $\pm$ 0.3	63.25 $\pm$ 0.3	<b>63.25</b> $\pm$ 0.3
1	69.42 $\pm$ 0.1	69.63 $\pm$ 0.1	69.67 $\pm$ 0.3	<b>69.80</b> $\pm$ 0.1
2	71.84 $\pm$ 0.3	71.84 $\pm$ 0.3	<b>72.14</b> $\pm$ 0.2	71.89 $\pm$ 0.1
3	73.34 $\pm$ 0.3	73.47 $\pm$ 0.1	<b>73.56</b> $\pm$ 0.2	73.47 $\pm$ 0.1
4	<b>74.53</b> $\pm$ 0.1	<b>74.53</b> $\pm$ 0.2	74.32 $\pm$ 0.2	74.53 $\pm$ 0.1
5	75.13 $\pm$ 0.1	<b>75.39</b> $\pm$ 0.2	75.19 $\pm$ 0.2	<b>75.39</b> $\pm$ 0.1

Table 5: **VOC07+12**. The results of adding top  $k\%$  most confident pseudo-labels for class, compared to the results of our method. *Top 20%*, *Top 30%*, *Top 40%* represent the methods where we choose to pseudo-label the most confident 20%, 30% and 40% pseudo-labels per class. *Ours* represent our method where we pseudo-label all the objects for which the network’s confidence is greater than 0.99.

## Engineering tricks to consider

### 5.1 Non-maximum suppression

We found the effect of non-maximum suppression (NMS) to be very important in all AL methods. Without applying NMS, active learning methods did not work better than a random sampling method. We hypothesize that this happens because if we do not apply NMS, the number of detected boxes is in the hundreds, so by sheer chance, some of them might have high acquisition scores. Considering that in a real-world scenario these boxes would be *killed* by NMS, we conclude that these boxes should not be used to compute an acquisition score. Thus, for every image, we apply NMS before proceeding with the computation of the acquisition score.

### 5.2 Balanced mini-batches

In the main paper, for every experiment, we force that half of the samples in a mini-batch are labeled. In this experiment, we evaluate the effect of varying the number of labeled samples in a mini-batch. In particular, we compare our results to having only half of the samples labeled, and a random approach. In order to be able to quantify the effect of balancing, we do all the experiments without adding pseudo-labels. We present the results in Table 6. We observe that our strategy of balancing the mini-batches so they contain an equal number of labeled and unlabeled samples, performs best by up to  $1pp$  in all AL cycles except the zeroth one, when it gets outperformed by  $0.12pp$  by the strategy where only a quarter of samples contain labels. We also observe that the strategy where we do only random sampling consistently reaches the worst results. In fact, in the zeroth AL cycle it gets outperformed by the balanced strategies by almost  $10pp$ . This can be explained by the fact that the

number of labeled samples (2,000) is much lower than the number of unlabeled samples (14,651), so in a mini-batch of size 32, in average, only 3.86 samples have labels. In some mini-batches, the number of labeled samples is 0, and thus the loss function becomes completely self-supervised. We observe that when the number of labeled samples increases (by labeling other images during AL stage), the overall performance increases, but it still lags behind the the balanced strategies.

Cycle	Random	Balanced quarter	Ours w/o $\tau_1$
0	53.66 $\pm$ 0.2	<b>63.37</b> $\pm$ 0.2	63.25 $\pm$ 0.2
1	67.39 $\pm$ 0.4	68.12 $\pm$ 0.2	<b>68.40</b> $\pm$ 0.3
2	69.90 $\pm$ 0.5	70.12 $\pm$ 0.6	<b>70.84</b> $\pm$ 0.8
3	71.38 $\pm$ 1.2	71.92 $\pm$ 0.3	<b>72.93</b> $\pm$ 0.3
4	73.53 $\pm$ 0.4	73.48 $\pm$ 0.3	<b>73.66</b> $\pm$ 0.2
5	74.30 $\pm$ 0.4	73.90 $\pm$ 0.4	<b>74.98</b> $\pm$ 0.5

Table 6: **VOC07+12**. Accuracy as a function of label/unlabeled sampling strategy. *Random* refers to random sampling from the entire dataset, *Balanced quarter* refers to having a quarter of labeled samples; *Ours* refers to half of the samples being labeled. Our balanced strategy outperforms the other two strategies. Note that in order to check only the effect of balancing, we do not add pseudo-labels during the training, thus, the results of *Ours* are different to the other papers where we add pseudo-labeling.

## References

- [1] Hamed Habibi Aghdam, Abel Gonzalez-Garcia, Antonio M. López, and Joost van de Weijer. Active learning for deep detection neural networks. In *International Conference on Computer Vision (ICCV)*, 2019.
- [2] William H. Beluch, Tim Genewein, Andreas Nürnberger, and Jan M. Köhler. The power of ensembles for active learning in image classification. In *Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [3] Sai Vikas Desai, Akshay Chandra Lagandula, Wei Guo, Seishi Ninomiya, and Vineeth N. Balasubramanian. An adaptive supervision framework for active learning in object detection. In *British Machine Vision Conference (BMVC)*, 2019.
- [4] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *International Journal in Computer Vision (IJCV)*, 88(2):303–338, 2010.
- [5] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning (ICML)*, 2016.
- [6] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *International Conference on Machine Learning (ICML)*, 2017.
- [7] Elmar Haussmann, Michele Fenzi, Kashyap Chitta, Jan Ivanecy, Hanson Xu, Donna Roy, Akshita Mittel, Nicolas Koumchatzky, Clement Farabet, and Jose M Alvarez. Scalable active learning for object detection. In *Intelligent Vehicles (IV)*, 2020.
- [8] Neil Houlsby, Ferenc Huszar, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *CoRR*, abs/1112.5745, 2011.
- [9] Jisoo Jeong, Seungeui Lee, Jeessoo Kim, and Nojun Kwak. Consistency-based semi-supervised learning for object detection. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
- [10] Chieh-Chi Kao, Teng-Yok Lee, Pradeep Sen, and Ming-Yu Liu. Localization-aware active learning for object detection. In *Asian Conference on Computer Vision (ACCV)*, 2018.
- [11] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2014.
- [12] Andreas Kirsch, Joost van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

- [13] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *International Conference on Learning Representations (ICLR)*, 2017.
- [14] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. *ICML Workshop on Challenges in Representation Learning*, 2013.
- [15] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *European Conference in Computer Vision (ECCV)*, 2014.
- [16] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European Conference on Computer Vision (ECCV)*, 2016.
- [17] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *IEEE Transactions in Pattern Analysis and Machine Intelligence (TPAMI)*, 41(8):1979–1993, 2019.
- [18] Avital Oliver, Augustus Odena, Colin Raffel, Ekin Dogus Cubuk, and Ian J. Goodfellow. Realistic evaluation of deep semi-supervised learning algorithms. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.
- [19] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.
- [20] Soumya Roy, Asim Unmesh, and Vinay P. Namboodiri. Deep active learning for object detection. In *British Machine Vision Conference (BMVC)*, 2018.
- [21] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations (ICLR)*, 2018.
- [22] Claude E. Shannon. A mathematical theory of communication. *Mobile Computing and Communications Review*, 5(1):3–55, 2001.
- [23] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015.
- [24] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- [25] Toan Tran, Thanh-Toan Do, Ian D. Reid, and Gustavo Carneiro. Bayesian generative active deep learning. In *International Conference on Machine Learning (ICML)*, 2019.
- [26] Donggeun Yoo and In So Kweon. Learning loss for active learning. In *Computer Vision and Pattern Recognition (CVPR)*, 2019.